

Robust supervised learning under uncertainty in dataset shift

Weihua Hu

Dept. of Computer Sci.
The University of Tokyo
hu@ms.k.u-tokyo.ac.jp

Issei Sato

Dept. of Compl. Sci. and Eng.
The University of Tokyo
sato@k.u-tokyo.ac.jp

Masashi Sugiyama

RIKEN
Dept. of Compl. Sci. and Eng.
The University of Tokyo
sugi@k.u-tokyo.ac.jp

Abstract

When machine learning is deployed in the real world, its performance can be significantly undermined because test data may follow a different distribution from training data. To build a reliable machine learning system in such a scenario, we propose a supervised learning framework that is explicitly robust to the uncertainty of dataset shift. Our robust learning framework is flexible in modeling various dataset shift scenarios. It is also computationally efficient in that it acts as a *robust wrapper* around existing gradient-based supervised learning algorithms, while adding negligible computational overheads. We discuss practical considerations in robust supervised learning and show the effectiveness of our approach on both synthetic and benchmark datasets.

1 Introduction

Supervised machine learning has been proven successful in many application fields including but not limited to natural language processing, computer vision and robotics. The vast majority of supervised learning research assumes that training data and test data are drawn from the same underlying distribution.

However, such an assumption can easily be broken when machine learning is deployed in the real world. Consider for instance, email spam filtering. The emails in the training data are not likely to be a representative of what is expected in the test data, because malicious spammers may avoid using typical spam words in the test stage, or a distribution of spammers may

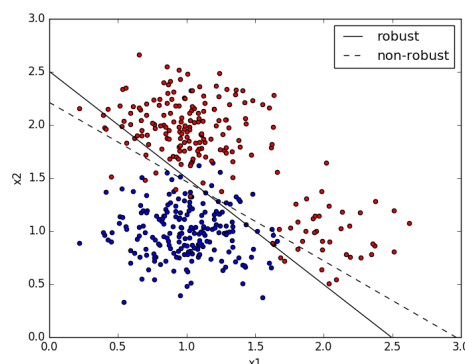


Figure 1: An illustration of robust classification and normal classification. The dots are training data, and the colors indicate class labels. The solid line and dashed line are the decision boundaries learned by robust learner and non-robust learner, respectively.

change. Furthermore, the process of data collection itself can induce sample selection bias. For instance, in the task of image classification, the collected training images may be biased toward those posted on the web, while the actual prediction should be made on daily images. In both scenarios, the performance of traditional machine learning techniques can be significantly degraded, making them unreliable for practitioners to use.

To build reliable machine learning systems in the wild, it is preferable to learn prediction functions that are explicitly robust to the uncertainty of changes in joint distributions from training data to test data, i.e. *dataset shift* (Quionero-Candela et al., 2009). Fig. 1 illustrates the idea by comparing robust classification and non-robust classification. Notice that the red data have two clusters and the training samples are biased toward one of the clusters. Compared to non-robust learner, the robust learner adjusts the decision boundary to prepare for the distribution shift to the other cluster. See Section 6 for the detailed setting.

In previous study, Bagnell (2005) considers robust supervised learning under distribution change with covariate shift assumption (Shimodaira, 2000), in which both training and test distribution share the same conditional distribution $p(y|x)$ while their marginal distributions, $p(x)$ and $q(x)$ are different. More specifically, they consider the following minimax game between a learner and an adversary: the adversary first shifts the test distribution from the training distribution within a given range measured by Kullback-Leibler (KL) divergence so as to maximize the total loss. The learner then minimizes the total loss. Their framework is promising and acts as a *robust wrapper* around a broad class of existing supervised learning algorithms. However, two major problems need to be overcome for the framework to be of practical use: 1) computational inefficiency and 2) inflexibility in modeling uncertainty of test distributions.

First, robust learning of Bagnell (2005) incurs huge computational overheads to the original non-robust learning problem: at every iteration, they require the cost-sensitive version of the original learning problem to be solved exactly.

Second, robust learning of Bagnell (2005) considers the test distribution $q(x)$ to change *arbitrarily* from training distribution $p(x)$ within a given range. Without any specific assumptions on dataset shift, the robust algorithm often provides overly pessimistic solutions (See Section 6.2) or be vulnerable to outliers. In practice, we may want to incorporate our assumptions on dataset shift to limit the class of dataset shift. For example, the class-balance change is a well-adopted assumption which assumes that only class prior $p(y)$ changes and class conditional $p(x|y)$ remains the same (Saerens et al., 2002). In many applications, the category y can be further divided into more refined subcategories (Ristin et al., 2015), e.g., the category ‘news’ may contain ‘sports’, ‘politics’ and ‘entertainment’ as subcategories. In such a case, we may assume the dataset shift to occur at subcategory levels. In other cases, conditions in which data are being collected may imply the potential dataset shift: data collected in the similar conditions tend to be shifted together. These assumptions on dataset shift enable the robust learning algorithms to focus on specific aspects of dataset shift. Hence, we can expect more meaningful solutions than pessimistic ones. A flexible robust learning framework is needed so that our assumptions on dataset shift can be incorporated into the modeling of the uncertainty set for test distributions.

The main contributions of our paper are as follows.

1. We propose a flexible robust learning framework that generalizes the robust learning framework of (Bagnell, 2005). More specifically, we use f -

divergences to constrain the distribution shift by the adversary. We also introduce weight sharing between data points to better model our assumption on dataset shift.

2. We show efficient gradient-based algorithms for a broad class of f -divergences: the KL divergence, Pearson (PE) divergence and *piece-wise linear (PWL) divergence*, which is a generalization of the total variation divergence. Our algorithm acts as a *robust wrapper* around existing gradient-based supervised learning algorithms, while adding negligible computational overheads to the original gradient calculation.

We also point out that the covariate shift assumption in Bagnell (2005) and Wen et al. (2014) can be removed and be readily extended into general dataset shift scenarios, where training distribution $p(x, y)$ and test distribution $q(x, y)$ are different.

The rest of the paper is organized as follows. Related work is summarized in Section 2, while our robust formulation is explained in Section 3. Robust learning for a broad class of f -divergences are studied in Section 4. Discussions regarding the computational complexity and other practical issues are made in Section 5. Experiments on both synthetic and real datasets are presented in Section 6 and a conclusion is drawn in Section 7.

2 Related work

Our work is most related to Bagnell (2005) and Wen et al. (2014). Both of them consider the minimax approach for learning under uncertain test distributions by adversarially reweighting the data. (Wen et al., 2014) also related covariate shift to model misspecification and proposed a way to judge whether covariate shift correction is necessary. Our work is a generalization of Bagnell (2005) in the sense that we use f -divergences instead of the KL divergence in modeling the uncertainty in test distributions.

Other minimax formulations of learning under uncertain test distribution include (Globerson and Roweis, 2006), (Liu and Ziebart, 2014) and (Chen et al., 2016). (Globerson and Roweis, 2006) learns a classifier robust to deletion of a subset of features, while (Liu and Ziebart, 2014) learns a classifier robust to unknown properties of the conditional label distribution, which was then extended by (Chen et al., 2016) into a regression setting.

Most existing researches on covariate shift correction involve density ratio estimation using both training and test data (Quionero-Candela et al., 2009). The estimated density ratio is then used to reweight the

training data to provide an unbiased estimate of the risk on the test data (Shimodaira, 2000). Our robust framework assumes that test data are not provided at training time and tries to be robust to the adversarial density ratio instead of estimating it.

3 Robust supervised learning under dataset shift uncertainty

In this section, we explain our robust formulation and provide a gradient-based optimization algorithm.

3.1 Generalized formulation of robust supervised learning

We start with formulating robust learning. Our formulation is a generalization of Bagnell (2005) by using f -divergences instead of the KL divergence and considering general dataset shift instead of covariate shift. Suppose we are given i.i.d. (independent and identically distributed) training samples $\{(x_1, y_1), \dots, (x_N, y_N)\}$, drawn from a training distribution on $\mathcal{X} \times \mathcal{Y}$ with density $p(x, y)$. We consider the dataset shift scenario, where the test density $q(x, y)$ is different from the training density $p(x, y)$. In particular, consider a scenario where the test distribution q can vary arbitrarily around the training distribution p within a given range measured by f -divergence D_f , defined as:

$$D_f[q(x, y) || p(x, y)] \equiv \mathbb{E}_{p(x, y)} \left[f \left(\frac{q(x, y)}{p(x, y)} \right) \right], \quad (1)$$

where $f(\cdot)$ is a convex function such that $f(1) = 0$. Let $g_\theta : \mathcal{X} \rightarrow \mathcal{Y}$ be a prediction function with parameter θ and $l(\hat{y}, y)$ is a loss between y and its prediction \hat{y} . Our learning objective then is to minimize the expected loss $\mathbb{E}_{q(x, y)}[l(g_\theta(x), y)]$ for the most adversarial test distribution q , which is formalized as follows.

$$\min_{\theta} \max_{q: D_f[q || p] \leq \delta} \mathbb{E}_{q(x, y)}[l(g_\theta(x), y)]. \quad (2)$$

Let $r(x, y) \equiv \frac{q(x, y)}{p(x, y)} \geq 0$. Eq. (2) is written as

$$\min_{\theta} \max_{r \in \mathcal{U}_f} \mathbb{E}_{p(x, y)}[r(x, y)l(g_\theta(x), y)], \quad (3)$$

where

$$\begin{aligned} \mathcal{U}_f \equiv \{r(x, y) \mid & \mathbb{E}_{p(x, y)}[f(r(x, y))] \leq \delta, \\ & \mathbb{E}_{p(x, y)}[r(x, y)] = 1, \\ & r(x, y) \geq 0, \forall (x, y) \in \mathcal{X} \times \mathcal{Y}\}. \end{aligned} \quad (4)$$

Note that all the expectations in Eqs. (3) and (4) are taken with respect to the training density $p(x, y)$, and thus can be approximated using training samples

$\{(x_1, y_1), \dots, (x_N, y_N)\}$. For notational convenience, we denote $r(x_i, y_i)$ and $l(g_\theta(x_i), y_i)$ as r_i and l_i , respectively. Also, we define vectors $\mathbf{r} \equiv (r_1, \dots, r_N)^\top$ and $\mathbf{l} \equiv (l_1, \dots, l_N)^\top$, where $^\top$ denotes the transpose. Then, we have

$$\min_{\theta} \max_{\mathbf{r} \in \widehat{\mathcal{U}}_f} \frac{1}{N} \sum_{i=1}^N r_i l_i, \quad (5)$$

$$\widehat{\mathcal{U}}_f = \left\{ \mathbf{r} \mid \frac{1}{N} \sum_{i=1}^N f(r_i) \leq \delta, \frac{1}{N} \sum_{i=1}^N r_i = 1, \mathbf{r} \geq 0 \right\}, \quad (6)$$

where an inequality constraint for a vector is applied in an elementwise fashion. We omit the regularization term in Eq. (5) for brevity. In practice, it is straightforward to add a regularization term.

3.2 Incorporating assumptions on dataset shift by weight sharing

The uncertainty set of Eq. (6) does not assume any model on the density ratio $r(x, y)$. As a result, the adversary simply puts larger weights to data points with larger losses, which often gives overly pessimistic solutions (See Section 6.2). It also makes the learned function vulnerable to outliers because the adversary can put large weights on them.

In practice, we can incorporate our assumptions on dataset shift to limit the class of dataset shift. To this end, we propose a simple weight sharing scheme to incorporate the assumptions. In the scheme, weights put by the adversary are shared among data points in the same *groups*. The *groups* are designed beforehand to reflect our assumptions on the distribution change: data points in the same group are assumed to be shifted together. For instance, the assumption of class-balance change is equivalent to sharing weights within the data points with the same class labels. In this case, each group corresponds to each label.

Let $\mathcal{C} = \{\mathbf{c}_1, \mathbf{c}_2, \dots, \mathbf{c}_K\}$ be a set of designed groups, where \mathbf{c}_k is a set of data indices belonging to the k -th group. We consider the following robust formulation.

$$\min_{\theta} \max_{\mathbf{w} \in \widehat{\mathcal{U}}'_f} \frac{1}{N} \sum_{k=1}^K n_k w_k \bar{l}_k, \quad (7)$$

$$\begin{aligned} \widehat{\mathcal{U}}'_f = \left\{ \mathbf{w} \in \mathbb{R}^K \mid \frac{1}{N} \sum_{k=1}^K n_k f(w_k) \leq \delta, \right. \\ \left. \frac{1}{N} \sum_{k=1}^K n_k w_k = 1, \mathbf{w} \geq 0 \right\}, \end{aligned} \quad (8)$$

where n_k is the number of data points in \mathbf{c}_k so that $\sum_{k=1}^K n_k = N$, w_k is the weight shared within data

points in \mathbf{c}_k , and \bar{l}_k is an average loss of data points in \mathbf{c}_k defined as follows:

$$\bar{l}_k = \frac{1}{n_k} \sum_{i \in \mathbf{c}_k} l_i. \quad (9)$$

We emphasize that the advantages of the weight sharing are in its flexibility and computational efficiency. It is flexible in that we can easily incorporate our assumptions such as those mentioned in the introduction. As we will see, it is computationally efficient in that it maintains the same order of efficiency as solving Eq. (5).

3.3 Gradient-based learning for the generalized formulation

We provide a gradient-based optimization algorithm. Our objective Eq. (7) is a minimax optimization of the form:

$$\min_{\theta} \max_{\mathbf{w} \in \hat{\mathcal{U}}'_f} L(\mathbf{w}, \theta). \quad (10)$$

Let $L^*(\theta) = \max_{\mathbf{w} \in \hat{\mathcal{U}}'_f} L(\mathbf{w}, \theta)$. Note that $L^*(\theta)$ is convex in θ if $l(g_\theta(x), y)$ is convex in θ because it is the maximum over a set of convex functions (Boyd and Vandenberghe, 2004). Danskin's theorem (Danskin, 2012) allows us to compute the gradient $\nabla L^*(\theta)$ as

$$\nabla L^*(\theta) = \frac{1}{N} \sum_{k=1}^N n_k w_k^* \nabla_{\theta} \bar{l}_k, \quad (11)$$

where

$$\mathbf{w}^* = \arg \max_{\mathbf{w} \in \hat{\mathcal{U}}'_f} L(\mathbf{w}, \theta). \quad (12)$$

Note that $\nabla L^*(\theta)$ does not depend on $\frac{\partial \mathbf{w}^*(\theta)}{\partial \theta}$. Hence, computing $\nabla L^*(\theta)$ only requires the numerical value of \mathbf{w}^* . Define $\mathbf{1}_m$ as an m -dimensional vector with all the elements being 1.

Theorem 1. *The uncertainty set $\hat{\mathcal{U}}'_f$ in Eq. (8) is a non-empty convex set.*

Theorem 1 can be proven by using the fact that $f(1) = 0$ and $f(\cdot)$ is convex.

Eq. (12) is a convex optimization because of Theorem 1 and linearity of $L(\mathbf{w}, \theta)$ in \mathbf{w} . Thus, any local optimum of Eq. (12) is a global optimum. As we will show, for a broad class of f -divergences, obtaining a numerical solution \mathbf{w}^* is very efficient, adding negligible computational overheads to the original gradient computation of $\nabla \bar{l}$ in Eq. (11). A gradient descent algorithm for our robust formulation is shown in Algorithm 1.

Algorithm 1 Batch optimization algorithm based on gradient descent

Require: $\{\alpha_t\}$: learning rate.

Require: $L(\theta, \mathbf{w})$: loss function.

Require: $\theta^{(0)}$: initial parameter.

$t \leftarrow 0$

while $\theta^{(t)}$ not converged **do**

$\mathbf{w}^* \leftarrow \arg \max_{\mathbf{w} \in \hat{\mathcal{U}}'_f} L(\mathbf{w}, \theta^{(t)}). \dots (*)$

$\theta^{(t+1)} \leftarrow \theta^{(t)} - \alpha_t \nabla_{\theta} L(\mathbf{w}^*, \theta^{(t)}).$

$t \leftarrow t + 1.$

end while

4 Robust learning in various f -divergences

In this section, we study using a broad class of f -divergences in our robust formulation, including the KL divergence, the PE divergence and the PWL divergence. We show that the optimization problem of Eq. (12) can be solved efficiently for these f -divergences. Computational overheads and the use of different f -divergences are analyzed in Sections 5.1 and 5.2, respectively.

4.1 KL divergence

When the f -divergence is the KL divergence, i.e., $f(x) = x \log x$, our formulation is the same as Bagnell (2005) except that we have the weight sharing. The optimization problem corresponding to Eq. (12) is

$$\begin{aligned} \max_{\mathbf{w}} \quad & \frac{1}{N} \sum_{k=1}^K n_k w_k \bar{l}_k, \\ \text{s.t.} \quad & \frac{1}{N} \sum_{k=1}^K n_k w_k \log w_k \leq \delta, \quad \frac{1}{N} \sum_{k=1}^K n_k w_k = 1, \quad \mathbf{w} \geq 0. \end{aligned} \quad (13)$$

Following Bagnell (2005), the optimal solution is obtained as

$$w_k^* = \frac{1}{Z(\gamma)} \cdot \exp\left(\frac{\bar{l}_k}{\gamma}\right), \quad 1 \leq k \leq K, \quad (14)$$

where γ is a scaler such that the first constraint of Eq. (13) holds with equality, and $Z(\gamma)$ is a normalizing constant in order to satisfy the second constraint of Eq. (13):

$$Z(\gamma) = \sum_{k=1}^K n_k \cdot \exp\left(\frac{\bar{l}_k}{\gamma}\right). \quad (15)$$

Inspired by Innuo (2010), we introduce a penalty pa-

parameter $\gamma' \geq 0$ and rewrite Eq. (13) as follows.

$$\begin{aligned} \max_{\mathbf{w}} \quad & \frac{1}{N} \sum_{k=1}^K n_k w_k \bar{l}_k + \frac{\gamma'}{N} \left(- \sum_{k=1}^K n_k w_k \log w_k \right) \\ \text{s.t.} \quad & \frac{1}{N} \sum_{k=1}^K n_k w_k = 1, \mathbf{w} \geq 0. \end{aligned} \quad (16)$$

γ' in Eq. (16) penalizes a large value of $\sum_{k=1}^K n_k w_k \log w_k$ and has the same role as δ in Eq. (13). The optimal solution of Eq. (16) is Eq. (14) with $\gamma = \gamma'$.

Note that for $\gamma' \rightarrow \infty$, the weights of Eq. (14) are uniform across the groups, and for $\gamma' \rightarrow 0$, the entire weight is allocated to a group with the largest average loss. If the loss function is convex in θ , it can be shown that there is an one to one correspondence between δ and γ' in the sense that for a given δ , there exists γ that gives the same solution θ^* of Eq. (10) and vice versa. Therefore, we can implicitly set δ by choosing an appropriate γ' .

Obtaining Eq. (14) is efficient and requires only $\mathcal{O}(K)$ computation.

4.2 PE divergence

We proceed to consider the case where the f -divergence is the PE divergence, i.e., $f(x) = (x - 1)^2$. The optimization problem corresponding to Eq. (12) is:

$$\begin{aligned} \max_{\mathbf{w}} \quad & \frac{1}{N} \sum_{k=1}^K n_k w_k \bar{l}_k, \\ \text{s.t.} \quad & \frac{1}{N} \sum_{k=1}^K n_k (w_k - 1)^2 \leq \delta, \frac{1}{N} \sum_{k=1}^K n_k w_k = 1, \mathbf{w} \geq 0. \end{aligned} \quad (17)$$

Since this is convex optimization, a point that satisfies the KKT conditions is the global optimum. In the preliminary experiments, we observed that for a small δ , it is quite unlikely for w_k to take a negative value due to the first quadratic penalty in Eq. (17). Therefore, we drop the inequality constraints $\mathbf{w} \geq 0$ of Eq. (17). Solving Eq. (17) then becomes simple:

$$\mathbf{w}^* = \sqrt{\frac{N\delta}{\sum_{k=1}^K n_k v_k^2}} \mathbf{v} + \mathbf{1}_K, \quad (18)$$

where \mathbf{v} is a K -dimensional vector such that

$$v_k = \bar{l}_k - \frac{1}{N} \sum_{k=1}^K n_k \bar{l}_k, \quad 1 \leq k \leq K. \quad (19)$$

Obtaining Eq. (18) is efficient and requires only $\mathcal{O}(K)$ computation.

Algorithm 2 Optimal solution \mathbf{w}^* for Eq. (21).

Require: $\delta, a, b, \alpha, \beta, N$

Require: $\bar{\mathbf{l}} = (\bar{l}_1, \dots, \bar{l}_K)^\top, \mathbf{n} = (n_1, \dots, n_K)^\top$.

Sort the elements of $\bar{\mathbf{l}}$, so that $\bar{l}_{t_1} \leq \bar{l}_{t_2} \leq \dots \leq \bar{l}_{t_K}$.

$\mathbf{w} \leftarrow \mathbf{1}_K$.

$k \leftarrow 1$.

$j \leftarrow K$.

$\Delta \leftarrow \frac{N\delta}{\beta - \alpha}$.

while $k < j$ **do**

if $n_{t_k}(1 - a) < \Delta$ **then**

$w_{t_k} \leftarrow a$,

$\Delta \leftarrow \Delta - n_{t_k}(1 - a)$.

$d \leftarrow n_{t_k}(1 - a)$.

while $(n_{t_j}(b - w_{t_j}) < d$ and $k < j$ **do**

$d \leftarrow d - n_{t_j}(b - w_{t_j})$,

$w_{t_j} = b$,

$j \leftarrow j - 1$.

end while

$w_{t_j} \leftarrow w_{t_j} + d/n_{t_j}$.

else

$w_{t_k} = 1 - \Delta/n_{t_k}$,

while $n_{t_j}(b - w_{t_j}) < \Delta$ and $k < j$ **do**

$\Delta \leftarrow \Delta - n_{t_j}(b - w_{t_j})$,

$w_{t_j} = b$,

$j \leftarrow j - 1$.

end while

$w_{t_j} \leftarrow w_{t_j} + \Delta/n_{t_j}$.

return \mathbf{w} .

end if

$k \leftarrow k + 1$.

end while

4.3 Piece-wise linear (PWL) divergence

We consider a specific member of f -divergences, which we call the *piecewise-linear (PWL) divergence* since the corresponding f is a PWL convex function defined as follows.

$$f(x) = \begin{cases} \max(\alpha(x - 1), \beta(x - 1)) & \text{if } a \leq x \leq b, \\ \infty & \text{otherwise,} \end{cases} \quad (20)$$

where $0 \leq a \leq 1 \leq b$ and $\alpha < \beta$. It is easy to see that the PWL divergence includes the total variation divergence ($f(x) = \frac{1}{2}|x - 1|$) as a special case. Also, setting $a = 0$ and δ to be sufficiently large, and removing the weight sharing, our robust formulation reduces to that of Wen et al. (2014) with the kernel width tending to 0.

The biggest advantage of using the PWL divergence for our robust formulation is that optimization of Eq. (12) can be solved very efficiently. Writing the

inner optimization problem explicitly, we have

$$\begin{aligned}
& \max_{\mathbf{w}} \frac{1}{N} \sum_{k=1}^K n_k w_k \bar{l}_k \\
& \text{s.t.} \quad \sum_{k=1}^K n_k w_k = N, \\
& \quad \frac{1}{N} \sum_{k=1}^K n_k \max(\alpha(w_k - 1), \beta(w_k - 1)) \leq \delta, \\
& \quad a \leq w_k \leq b, \text{ for } 1 \leq k \leq K.
\end{aligned} \tag{21}$$

The optimization of Eq. (21) is a linear program (LP). However, the second inequality constraint in Eq. (21) is expanded to an exponential number of linear constraints, and therefore, are generally intractable for LP solvers. However, surprisingly, Eq. (21) can be solved very efficiently without using LP solvers.

Algorithm 2 states how to obtain \mathbf{w}^* . It assigns a weight to each group according to its ranking of the average loss value: groups with higher average losses are given higher weights. Conceptually, the algorithm is transferring weights from groups with small average losses to groups with large average losses. The total transferred weights is at most $\Delta = \frac{N\delta}{\beta-\alpha}$. Therefore, the essential parameter for solving (21) is $\Delta \equiv \frac{N\delta}{\beta-\alpha}$, a and b . The dominant computation of Algorithm 2 involves the sorting of K elements, $\bar{l}_1, \bar{l}_2, \dots, \bar{l}_K$, which requires $\mathcal{O}(K \log K)$ time.

Theorem 2. *Algorithm 2 gives an optimal solution of (21)*

Proof. In LP, a local maxima is a global maxima. Thus, it is sufficient to show that Algorithm 2 provides a solution that is locally maximal. However, this is obvious since changing the elements of \mathbf{w}^* locally can only decrease the objective. \square

5 Discussions

In this section, we first discuss the computational overheads incurred by introducing the adversary. We then discuss qualitatively the use of different f -divergences and other practical considerations.

5.1 Computational overheads

In Section 4, we showed that the optimization of Eq. (12) can be solved efficiently for a broad class of f -divergences: $\mathcal{O}(K)$ for the KL and PE divergences, and $\mathcal{O}(K \log K)$ for the PWL divergence. Let T_g and T_{grad} be the computational time to evaluate $l(g_\theta(x), y)$ and $\nabla l(g_\theta(x), y)$ for one data point (x, y) , respectively. Then, the total computational time to make one parameter update in Algorithm 1

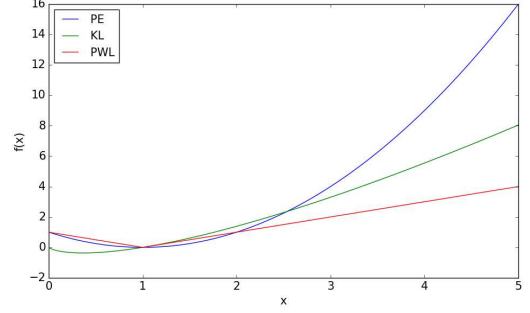


Figure 2: Comparison between different f -divergences.

is $\mathcal{O}(N \cdot (T_g + T_{\text{grad}}) + K)$ for the KL and PE divergences, while $\mathcal{O}(N \cdot (T_g + T_{\text{grad}}) + K \log K)$ for the PWL divergence. On the other hand, for the non-robust setting, one parameter update and the loss evaluation together take $\mathcal{O}(N \cdot (T_g + T_{\text{grad}}))$. We see that for most scenarios, $\mathcal{O}(K)$ or $\mathcal{O}(K \log K)$ adds negligible computational overheads to the non-robust problem.

(Wen et al., 2014) has similar formulation as ours and also solves the optimization problem by gradient descent. However, at each iteration, the adversary needs to solve a LP of size N with N constraints. The computation time for this is $\mathcal{O}(N^3)$ (Boyd and Vandenberghe, 2004) which adds a huge computational overhead for large data size N .

5.2 Qualitative comparison between different f -divergences

For $1 \leq x$, the f function for the PE, KL and PWL divergences are x^2 , $x \log x$ and $x \cdot \mathbf{1}_{\infty}\{x \leq b\}$, respectively, where $\mathbf{1}_{\infty}\{\cdot\}$ is 1 if the condition is true, and ∞ otherwise. This is depicted in Fig. 2. We see that the PE divergence incurs particularly harsh penalization to the large deviation of weights from 1, while the penalization is more flattened for the KL and PWL divergences. The choice of the f -divergences should be based on our inductive bias on dataset shift. When we anticipate abrupt (resp. mild) change in the data distribution, the PWL (resp. PE) divergence may be preferable.

5.3 Deciding the volume of the uncertainty set

The volume of the uncertainty set of Eq. (8) is essentially controlled by δ . How can we decide δ in practice? There is a clear trade-off between the robustness of the prediction function and its performance on the training distribution: the more robust the prediction function is, the worse its performance becomes on the training distribution. Therefore, we argue that δ should be as large as possible to ensure robustness of the prediction function, yet its performance on the

training distribution should be no worse than a pre-defined threshold. This is to ensure that the performance would not be so bad when there is actually no dataset shift. Following this principle, we can decide the volume of the uncertainty set by cross validation.

5.4 PE divergence and effective sample size

The PE divergence is directly related to the effective sample size N_{eff} of the weighted dataset (Priestley, 1981), which is defined using the weights $\{r_i\}_{i=1}^N$ as follows.

$$N_{\text{eff}} \equiv \frac{\left(\sum_{i=1}^N r_i\right)^2}{\sum_{i=1}^N r_i^2}. \quad (22)$$

Define the effective sample proportion as $\frac{N_{\text{eff}}}{N}$. Consider $\{r_i\}_{i=1}^N$ such that the effective sample proportion is larger than or equal to a given threshold s_{eff} .

$$N_{\text{eff}} = \frac{N^2}{\sum_{i=1}^N (r_i - 1)^2 + N} \geq s_{\text{eff}} \cdot N, \quad (23)$$

which is equivalent to upper bounding the empirical estimate of the PE divergence as follows:

$$\widehat{\text{PE}}(p||q) \leq \frac{1 - s_{\text{eff}}}{s_{\text{eff}}}. \quad (24)$$

Letting $\delta \equiv \frac{1 - s_{\text{eff}}}{s_{\text{eff}}}$, we see that δ , which controls the volume of the uncertainty set in our robust formulation, directly relates to the effective sample proportion of the weighted data. This interpretation makes it easy for us to choose δ such that the learned prediction function is robust enough yet is learned based on enough samples.

6 Experiments

In this section, we report experimental results. In Sections 6.1 and 6.2, we use a synthetic data to illustrate the use of the different f -divergences and the effectiveness of the weight sharing. In Section 6.3, we evaluate our approach using benchmark datasets.

6.1 Experimental comparison between different f -divergences

We illustrate the use of different f -divergences on a synthetic binary classification problem. The two class-conditional distributions are

$$\begin{aligned} p(x|y=1) &= \mathcal{N}(x \mid (1, 1)^\top, (0.25)^2 \mathbf{I}_2), \\ p(x|y=0) &= 0.8 \cdot \mathcal{N}(x \mid (1, 2)^\top, (0.25)^2 \mathbf{I}_2) \\ &\quad + 0.2 \cdot \mathcal{N}(x \mid (2, 1)^\top, (0.25)^2 \mathbf{I}_2), \end{aligned}$$

where $\mathcal{N}(\cdot|\boldsymbol{\mu}, \boldsymbol{\Sigma})$ denotes the multivariate Gaussian distribution with mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$, and \mathbf{I}_2 is an identity matrix of size 2. We created a synthetic dataset by drawing 200 samples for each class. We wrapped logistic regressions by our robust formulation. We did not implement weight sharing for simplicity. We fit our robust models using three different f -divergences and qualitatively compared the weights put by the adversaries. We set $\delta = 0.1$ for the PE divergence, $\Delta = 0.2 \cdot 400$, $a = 0.1$, $b = 5$ for the PWL divergence, and $\gamma = 1.5$ for the KL divergence. Fig. 3 illustrates the result, where each point in the graph corresponds to a data point, and its x (resp. y)-axis indicates the weights put by the PE (resp. KL or PWL) divergence. We see that using the KL and PWL divergences would result in more peaky weights allocation compared to the use of the PE divergence. This coincides with our analysis in Section 5.2.

6.2 Weight sharing

In this section, we use the synthetic data in Section 6.1 to illustrate the weight sharing proposed in Section 3.2. Recall that the weight sharing is introduced to incorporate our inductive bias on dataset shift into modeling the uncertainty set of test distributions. For our robust model, we used the KL divergence and set $\gamma = 0.02$. Fig. 4 shows the decision boundaries learned without weight sharing. The dots in the figure are data points and their sizes (areas) are proportional to the weights allocated by the adversary. We see that learned decision boundary is overly pessimistic, putting large weights to misclassified points. Now let us assume that dataset shift occurs at cluster levels. We model our assumption on dataset shift by clustering the data points into three groups and impose weight sharing within each group. The result is shown in Fig. 5. By constraining the adversaries though weight sharing, we obtain a more reasonable decision boundary that reflects our assumption on dataset shift.

6.3 Experiments on real datasets

We evaluate the robustness of our model on two benchmark datasets: the MNIST dataset and 20News group dataset. For MNIST, the dimensionality was reduced to 10 using principal component analysis. For 20News group, we removed stop words and retained 2000 most frequent words. We then removed documents with less than 10 words and used Latent Dirichlet Allocation (Blei et al., 2003) to reduce the dimensionality to 20 and 50. We randomly divided the datasets into training, validation and test datasets. For MNIST, the split ratio is 5: 1: 1, while for 20News group, the split ratio is 0.64: 0.16: 0.2.

We used logistic regressions with L2 norm regulariza-

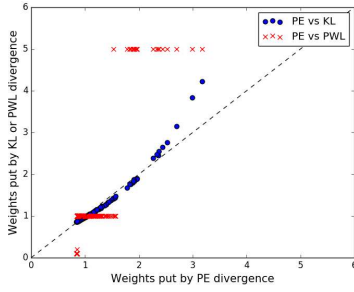


Figure 3: Comparison of weights put by different f -divergences.

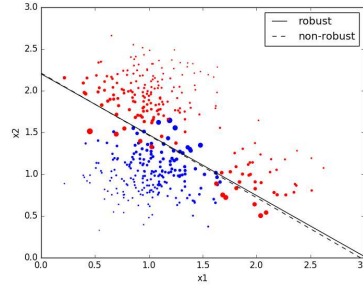


Figure 4: Robust classification without weight sharing.

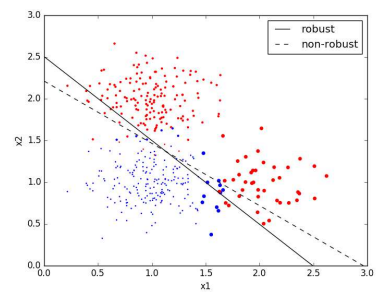


Figure 5: Robust classification with weight sharing among clusters.

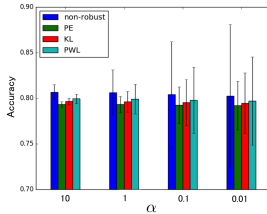


Figure 6: Results on MNIST.

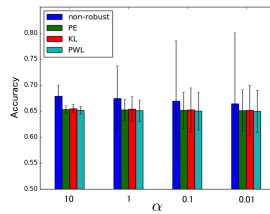


Figure 7: Results on 20Newsgroup.

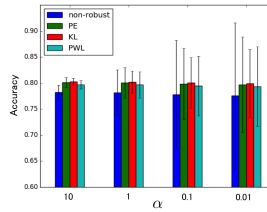


Figure 8: Results on MNIST using biased training data.

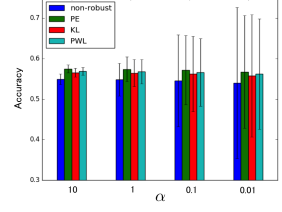


Figure 9: Results on 20Newsgroup using biased training data.

tion and selected the hyper-parameter using the validation datasets. In experiments, we focus on building classifiers robust to the class-balance shift. Thus, the weights are shared among data points with the same labels. In practice, we can loosen the assumption by considering weight sharing among more refined groups.

We selected δ (or γ in the KL divergence case) by the scheme introduced in Section 5.3. More specifically, we searched the biggest δ (or smallest γ) such that the accuracies on the validation sets are higher than given accuracy thresholds. We set $a = 0.5$ and $b = 3$ for the PWL divergence.

At the test stage, we assume that class priors would change. We model this by weighting test data points according to their class labels. The weights for the class labels are assumed to be drawn from a symmetric Dirichlet distribution with parameter $\alpha \mathbf{1}_K$, where K is the number of groups. As α gets smaller, the samples from the distribution have higher variance around the mean $\frac{1}{K} \mathbf{1}_K$, which corresponds to the higher uncertainty in test distributions.

In Figs. 6 and 7, we report weighted accuracies averaged over 10000 test datasets whose class priors are drawn from the Dirichlet distributions. The accuracy thresholds were set to 0.79 and 0.65 for MNIST and 20Newsgroup, respectively. For 20Newsgroup, we used 20-dimensional feature vectors and considered classifications over the last four labels. On average, the non-robust classifier outperforms the robust ones. However, the variance of accuracies using non-robust clas-

sifiers quickly gets larger as the test distributions become more uncertain. On the other hand, robust classifiers show more stable performance, making them reliable to use in practice.

We also created biased training and validation datasets by subsampling $1/i$ of data points for the i -th label. Figs. 8 and 9 report the averages and standard deviations of the weighted accuracies. For 20Newsgroup, we used 50-dimensional feature vectors. We set the accuracy thresholds to be 0.79 and 0.59 for MNIST and 20Newsgroup, respectively. We see that our robust algorithms perform better on average and is more stable than non-robust algorithms.

7 Conclusion

In this paper, we present a robust supervised learning framework that is both flexible in modeling the uncertainty in test distributions, and computationally efficient to learn. We discussed practical considerations in designing the uncertainty set of test distributions and empirically showed the effectiveness of our approach.

As machine learning algorithms are becoming more prevalent in the real world, their reliability has become an important issue. To achieve the reliability, we think there are two main approaches: 1) to secure the reliability *before* the systems are deployed and 2) to adapt to environmental change *after* the systems are deployed. In this paper, we tackled the problem of dataset shift from the first approach. It is our future work to tackle the problem from the second approach.

References

- J Andrew Bagnell. Robust supervised learning. In *Proceedings of the national conference on artificial intelligence*, volume 20, page 714. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2005.
- David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022, 2003.
- Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.
- Xiangli Chen, Mathew Monfort, Anqi Liu, and Brian D Ziebart. Robust covariate shift regression. In *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics*, pages 1270–1279, 2016.
- John M Danskin. *The theory of max-min and its application to weapons allocation problems*, volume 5. Springer Science & Business Media, 2012.
- Amir Globerson and Sam Roweis. Nightmare at test time: robust learning by feature deletion. In *Proceedings of the 23rd international conference on Machine learning*, pages 353–360. ACM, 2006.
- Innuo. Regularized Minimax for Robust Learning, 2010. URL <https://mlstat.wordpress.com/tag/danskins-theorem/>.
- Anqi Liu and Brian Ziebart. Robust classification under sample selection bias. In *Advances in Neural Information Processing Systems*, pages 37–45, 2014.
- Maurice Bertram Priestley. Spectral analysis and time series. 1981.
- Joaquin Quionero-Candela, Masashi Sugiyama, Anton Schwaighofer, and Neil D Lawrence. *Dataset shift in machine learning*. The MIT Press, 2009.
- Marko Ristin, Juergen Gall, Matthieu Guillaumin, and Luc Van Gool. From categories to subcategories: large-scale image classification with partial class label refinement. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 231–239, 2015.
- Marco Saerens, Patrice Latinne, and Christine Decaestecker. Adjusting the outputs of a classifier to new a priori probabilities: a simple procedure. *Neural computation*, 14(1):21–41, 2002.
- Hidetoshi Shimodaira. Improving predictive inference under covariate shift by weighting the log-likelihood function. *Journal of statistical planning and inference*, 90(2):227–244, 2000.
- Junfeng Wen, Chun-Nam Yu, and Russell Greiner. Robust learning under uncertain test distributions: Relating covariate shift to model misspecification. In *ICML*, pages 631–639, 2014.